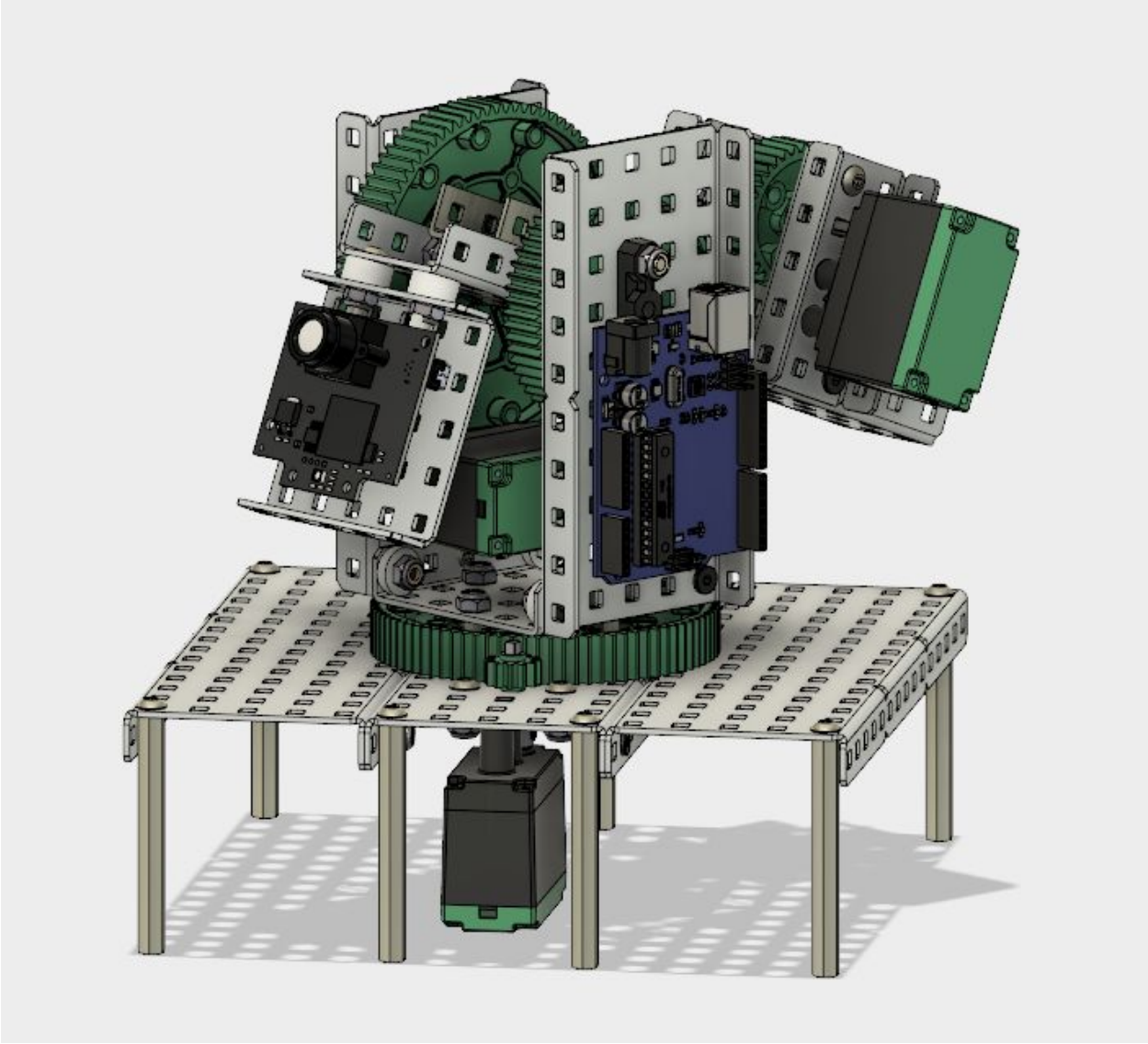
Pixy CMUcam5 For Vex Robotics



Noah Boursier **May 2018**

Table of Contents:

Problem Description

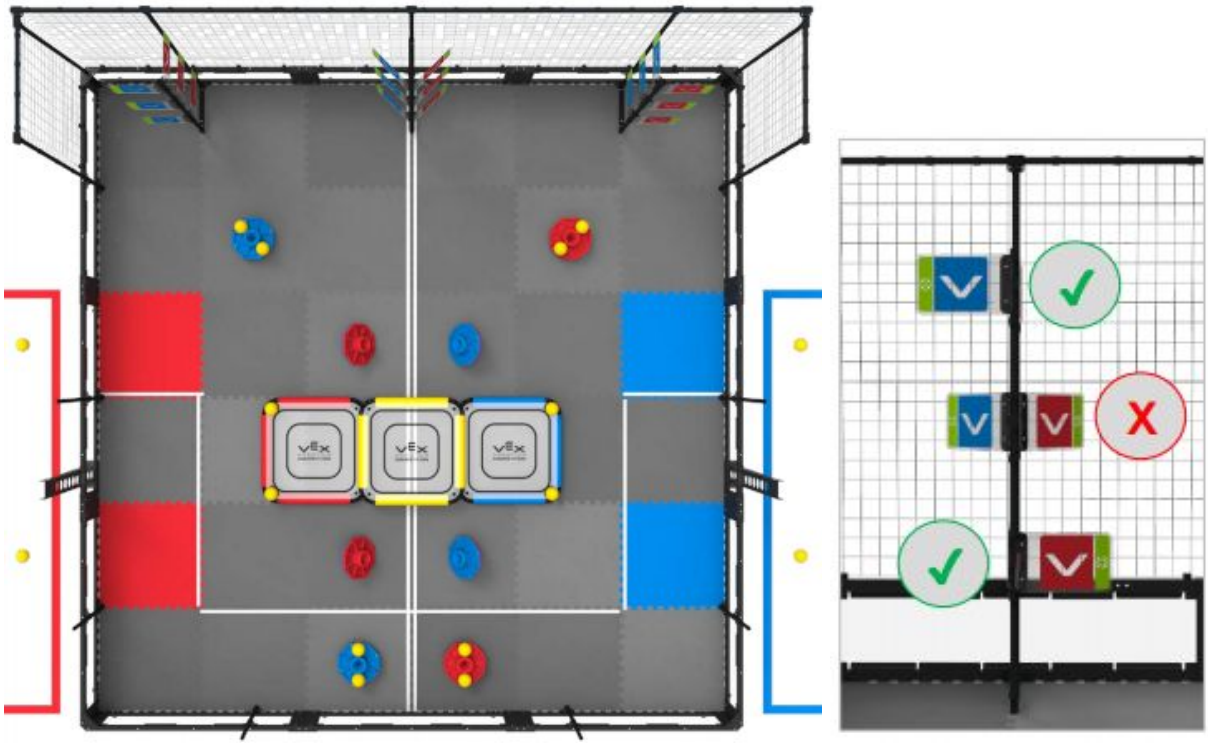
In the 2018-2019 Vex Robotics challenge, Turning Point, robots are expected to be able to shoot balls in order to turn rigid ‘flags’ so that their team color is shown. Also expected this competition robotics season is the widely anticipated new set of Vex sensors, motors and microcontroller, as a set known as V5. The problem is that Vex will not be releasing the new ‘vision sensor’ to the public until august, which is already three months into the competition season. The idea for this season is to be able to use the Vex vision sensor to enable robots to automatically locate and shoot flags of the right color This would mean that no matter where the robot is on the field they should be able to do a projectile calculation and figure out how far a flag is, whether it is the right color, and whether they are aligned to fire a successful shot. In order to see whether this is really possible next year we would need to run some tests, but because V5 is coming out in august we will not be able to run these tests until just weeks before our first competition. It is clear to me that we need to find some way to test whether it is feasible with the new Vex vision sensor to locate and aim a launcher at specific targets long before august comes around. I will be doing this by using a sensor similar that the new Vex vision sensor is based off of and by building a test launcher out of vex parts.

Research Summary

In previous years only one Vex team has used a Pixy camera on their robot as a sensor, the team used it to align with a pole that they needed to hang their robot on, which was hard for the driver to do without being directly behind the pole. I'm not entirely sure if they even used the sensor for autonomous alignment, but they could've. The Pixy camera that they used has an output that sends and analog signal depending on where the requested detected object is in its field of view. So as when the robot first sees the pole it will begin alignment with it by moving the robot so that the pole is in the center of the Pixy's vision.

In regard to using the a Pixy camera to align something to fire at objects, there are a few non-vex examples that I've seen, all hobby jobs, typically pretty crude and with flat projectile motion or no unique code other than what can be found in the online library. On top of this their view range (excluding the one in the left picture below) has has been too limited for practical use out on the Vex competition field.

As soon as I learned about the new Vex game I did a fair amount of research into what would need to happen for automatic aim to be a reality. Depicted is a top-down view of the Vex Turning Point competition match field.



At the top are the poles with flags, as you can see there are three different heights for the flags, the bottom can be flipped manually by a robot ramming it, but the upper two must be toggled with balls. The total area of the field is 144 square feet with the length of each side being twelve feet. Assuming that a robot should not attempt to hit a flag at an angle more severe than 25 degrees the maximum distance that a robot would have to fire a ball can be calculated also accounting for the placement of the launcher inside of the robot:

Max Flat Distance = (12-2)/cos(25°) = approx. 10.1 feet.

Then using pythagorean theorem to factor in the height of the uppermost flag:
(subtracting a foot for estimated camera position on a robot)

Max Distance = (sqrt((10.1*12)^2 + (46.3-(6/2)-12)^2))/12 = approx. 10.4 feet.

Combined Expression:

$$\sqrt{((11-(P\,ixy\ dist.\ to\ rear\ of\ bot))/cos(25°))^2\ +\ (43.3-(P\,ixy\ height\ on\ bot))^2}\ /\ 12$$

So the Pixy camera should be able to identify a target that has the dimensions of a flag (9.9” by 6”) at a maximum of 10.4 feet if we will only fire at a maximum angle of 25°, 11.8 feet for 30°, 12.5 feet for 35°, 13.3 feet for 40°, and 14.4 feet for 45°.

The another possible issue is how rapidly the Pixy will update, whether its tick rate be fast enough for speedy and accurate aiming during an actual competitive match. Online the Pixy boasts a whopping 50 frames per second, this is probably more than enough to aim in real time. Testing will be done to confirm this.

Next up is the option to use color codes:

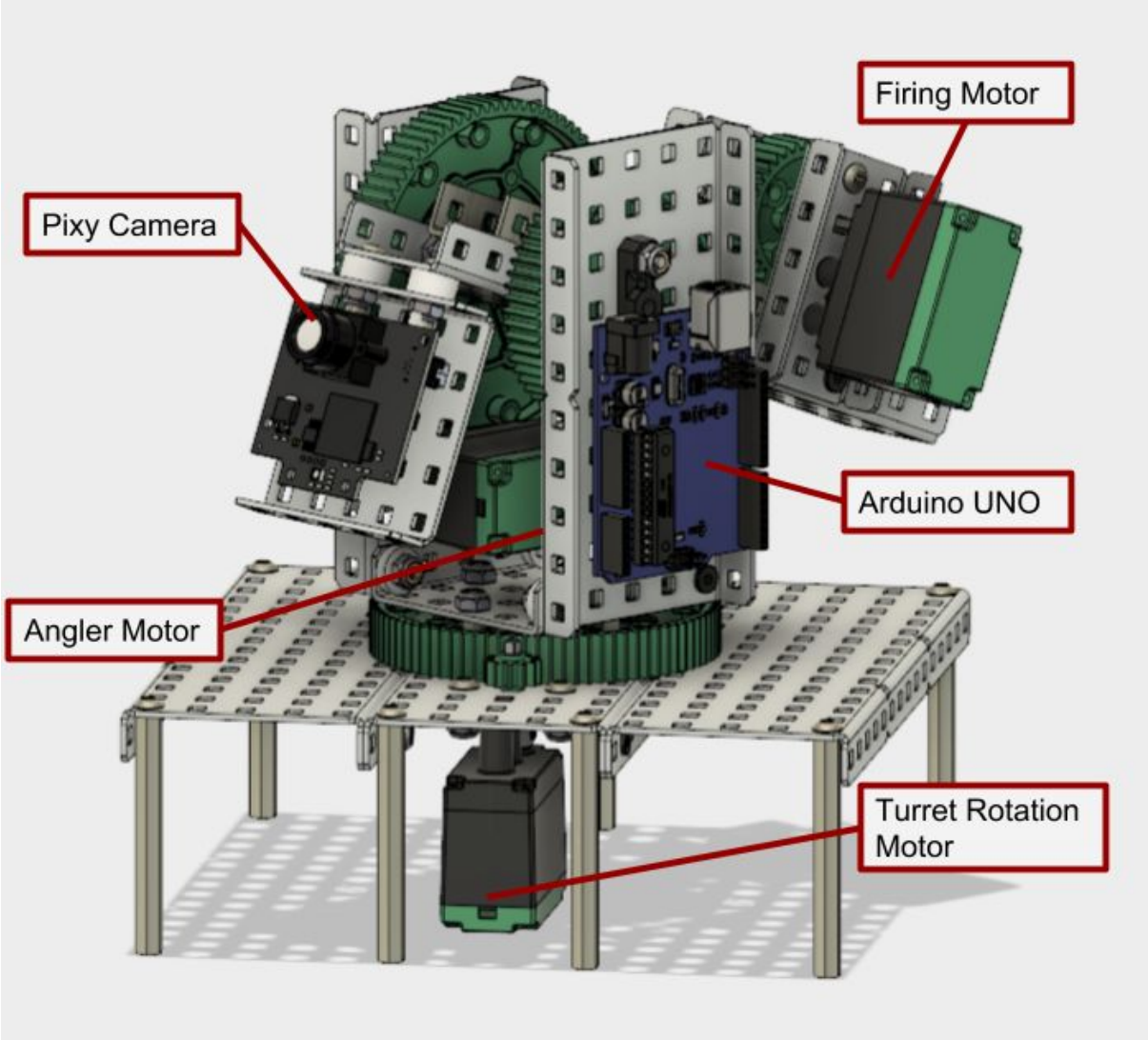
(Example from www.cmucam.org)

The Pixy camera has the ability to use color codes as both a more certain way to confirm that what the camera is recognizing as a target is actually a target and not just a background element that happens to have the same color. The Pixy can also return the angle of a color code object, though I'm not sure that this application of the Pixy will have much use for that information. But using a color code for say, red and green to identify flags seems like a good way to ensure that the robot is not targeting background elements. But if the green strip on the flag is too far away I'm afraid the Pixy may have a hard time identifying the green alongside the red or blue, leading to a flag detection failure.

One more issue with using a Pixy is that you may need to do a light calibration. If this could be done automatically, perhaps with known colors mounted within the Pixy's view on the robot so that it calibrates to the field's lighting right as the match begins. I'm not sure if the color signatures can be set in by the microcontroller or if they can only be set manually though. So far I've only seen Pixy's set color signatures manually, though it should be possible to do it automatically.

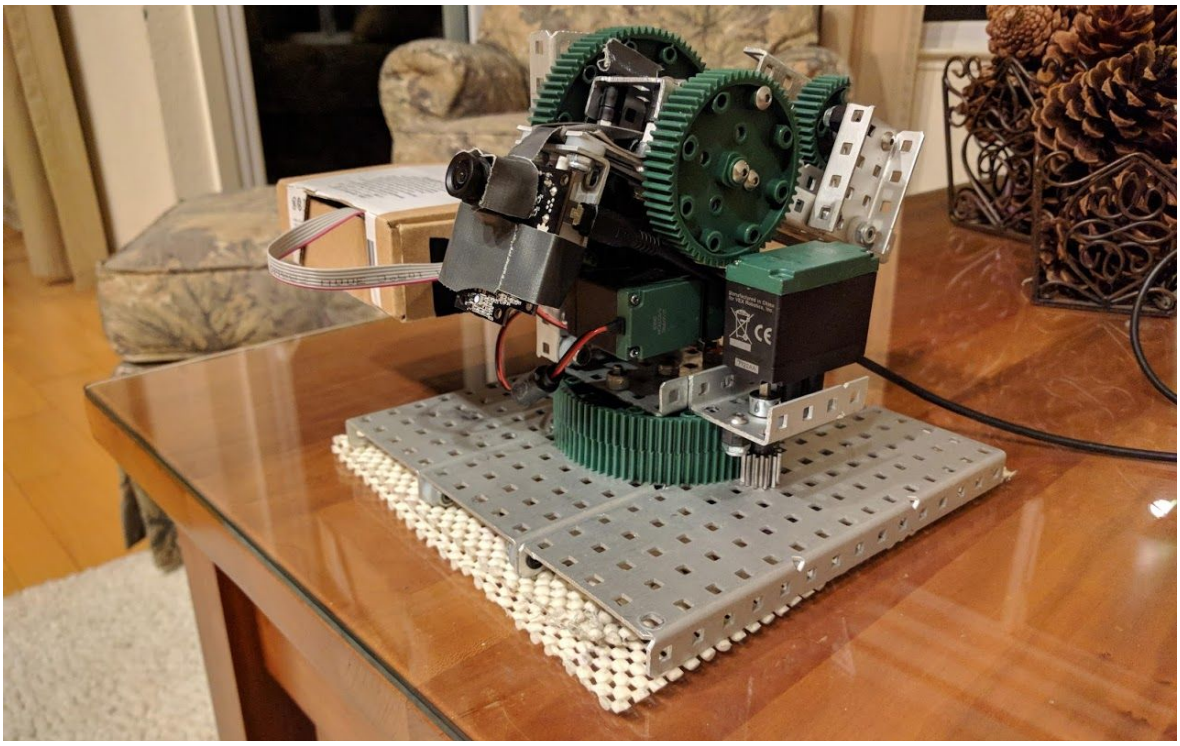
Solution Summary / Experiment Design

To test the aiming capabilities of the Pixy camera I decided to design a launcher with aiming system that could change where the Pixy camera is looking as it moves about. The turret base has 360 degrees of freedom (not considering the turret rotation motor cables), and the launcher itself has 45 degrees of freedom when angling up and down. The launcher is a simple linear slide with rack and slip gear. It uses rubber bands to propel the ‘puncher’ forward into the ball after the slip gear has rotated so that the part with no teeth is on the rack, allowing the rack to fly forward.

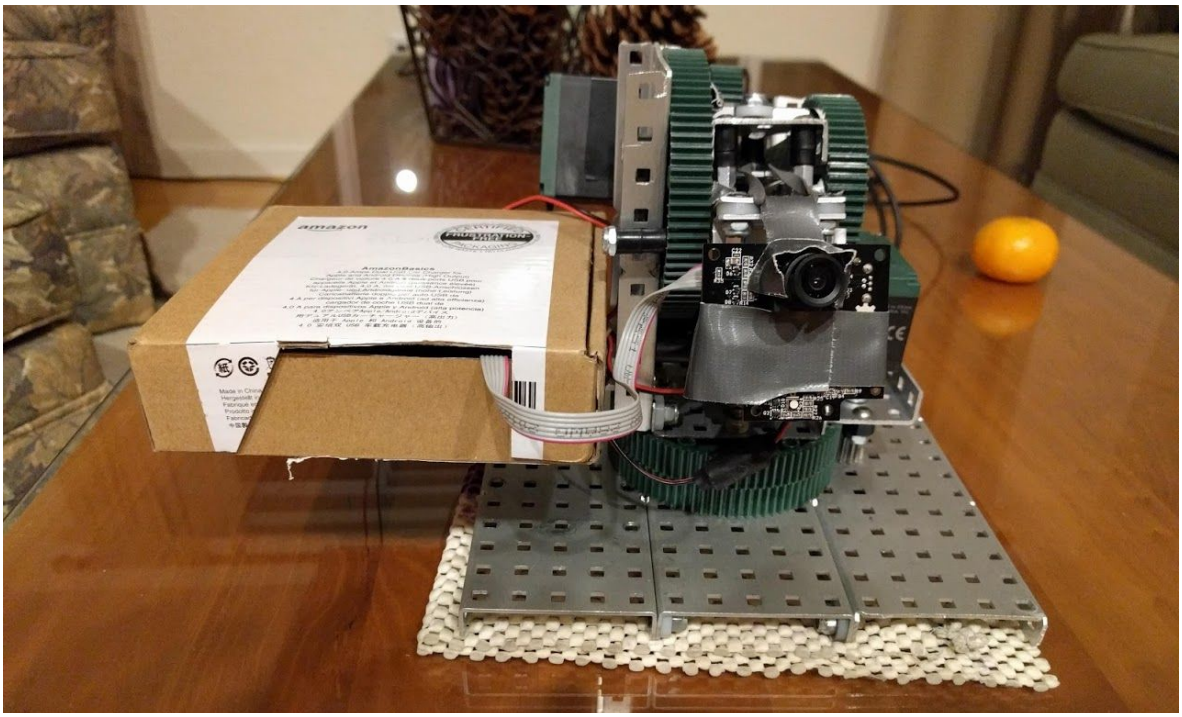


I decided the model a design for the turret in Autodesk Fusion 360 so that when I went in to get parts to actually build it I would have list of materials necessary at the ready (see page 11).

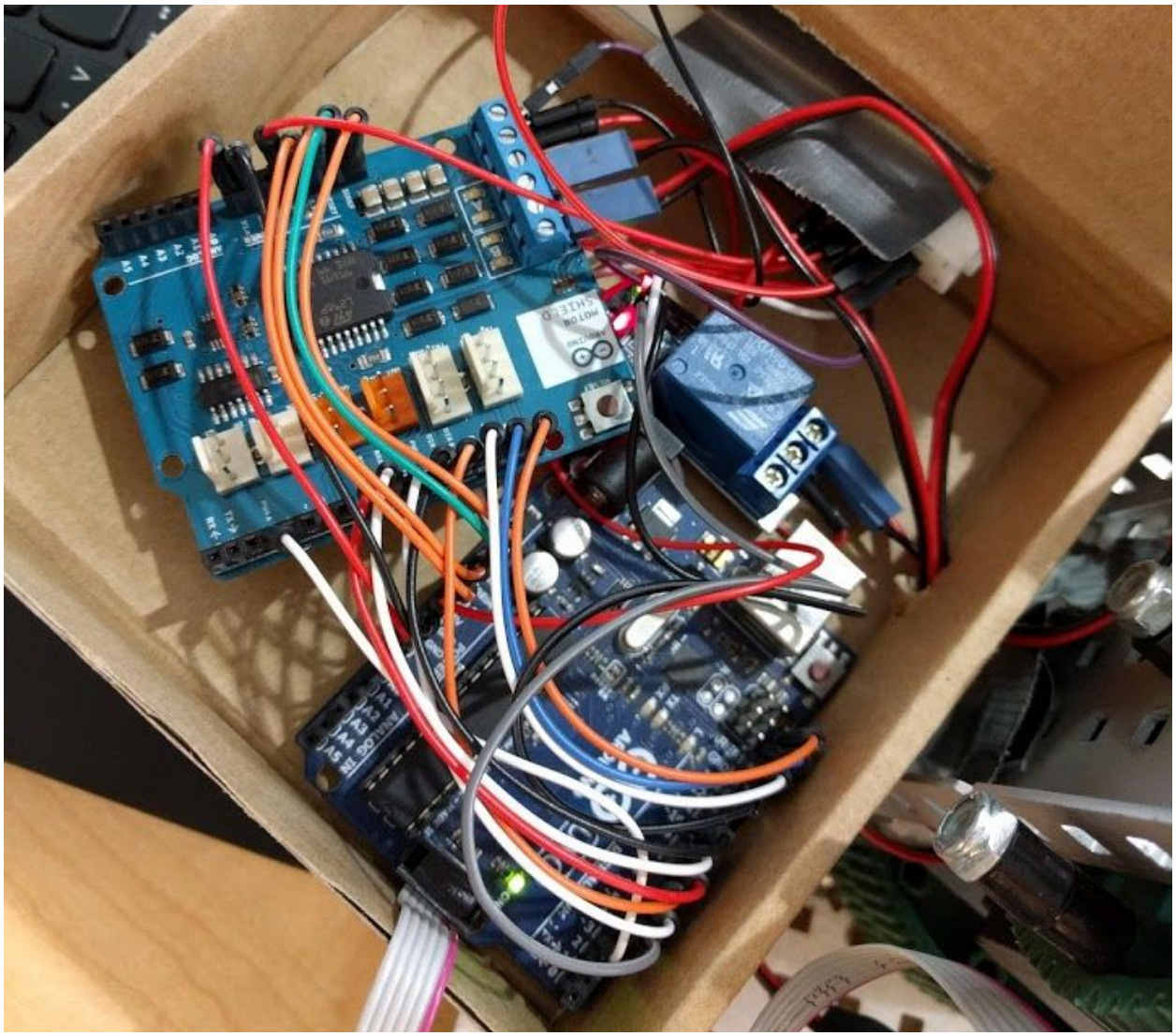
I ended up tweaking the design to allow unlimited rotation of the turret by moving the lower ‘turntable’ motor onto the turret itself, so that it spins a pinion along a fixed gear under the turret to cause the turret to rotate. You can see it in the picture below.



I placed a mesh pad underneath the base to increase friction because the base would spin instead of the turret when I tried to rotate due to the turret’s higher inertia.



I had to jump the Arduino motor shield to the Arduino UNO with wires instead of with the built in pins because of a conflict between the Pixy camera initialization and the motor shield relay control pins. I spent at least two hours figuring that out. Also because the motor shield only has two relays needed a third to operate the puncher. Seen below.



Fire danger? I don't know what you're talking about..

Because that mess of wires looks so hideous I found a box to hide it all away and mounted it to the turret so that it could rotate indefinitely without pulling any wires out.

Now onto the code. I'm using Arduino (the language for Arduino microcontrollers) which is basically Java, nothing too fancy. The turret only uses one sensor, the Pixy camera. What's cool about the Pixy is that you can actually use extremely weak (processing wise) microcontrollers with it, like an Arduino UNO with it. How could an Arduino possibly have the power to process images you ask? It doesn't, the Pixy has built in

chips that process the images automatically and then send only what's important back to the Arduino; how many objects are detected, their heights and widths, locations, colors, etc. This enables me to easily program the turret. Here's some of the major functions that my code runs through.

Helper function that makes it easy to write to the rotation motor for aiming:

```
void rot(int input) {
if (input < 0) {
  digitalWrite(rotDir, LOW);
}
else {
  digitalWrite(rotDir, HIGH);
}
  digitalWrite(rotBrake, LOW);
  analogWrite(rotSpeed, abs(input));
}
```

Finds the index of the biggest object of specified color in view:

```
for (j=0; j<blocks; j++)
{
  if (pixy.blocks[j].width * pixy.blocks[j].height > pixy.blocks[biggest].width * pixy.blocks[biggest].height) {
    biggest = j;
  }
}
```

Aiming code, proportional to object height and width as well as distance off target, meaning the motor will go faster the further off target the detected object is:

```
if (pixy.blocks[biggest].width > 10 && pixy.blocks[biggest].height > 10) {
  if (pixy.blocks[0].x > (150 + pixy.blocks[biggest].width/2)) {
    rot(-1*(35+((220*(pixy.blocks[biggest].x - (150 + pixy.blocks[biggest].width/2)))
    /(150 + pixy.blocks[biggest].width/2))));            // proportional speed control.
  }
  if (pixy.blocks[0].x < (150 - pixy.blocks[biggest].width/2)) {
    rot(35+((220*((150 - pixy.blocks[biggest].width/2) - pixy.blocks[biggest].x))
    /(150 - pixy.blocks[biggest].width/2)));
  }
  if (pixy.blocks[biggest].x < (150 + pixy.blocks[biggest].width/2) &&
  pixy.blocks[biggest].x > (150 - pixy.blocks[biggest].width/2)) {
    rot(0);
  }
  if (pixy.blocks[biggest].y > (90 + pixy.blocks[biggest].height/2)) { // 200 / 2 = 100 - 8 for error = 92
    tilt(-1*(35+((220*(pixy.blocks[biggest].y - (90 + pixy.blocks[biggest].height/2)))
    /(90 + pixy.blocks[biggest].height/2))));
  }
  if (pixy.blocks[biggest].y < (90 - pixy.blocks[biggest].height/2)) {
    tilt(35+((220*((90 - pixy.blocks[biggest].height/2) - pixy.blocks[biggest].y))/
    (90 - pixy.blocks[biggest].height/2)));
  }
  if (pixy.blocks[biggest].y < (90 + pixy.blocks[biggest].height/2) &&
  pixy.blocks[biggest].y > (90 - pixy.blocks[biggest].height/2)) {
    tilt(0);
  }
}
```

Of course this isn't everything but you get the idea.

Parts List

Metal

- 15 hole long 5 hole wide (7.5” x 2.5”) Vex C-Channels x3
- 10 hole long 5 hole wide (5” x 2.5”) Vex C-Channels x2
- 5 hole long 5 hole wide (2.5” x 2.5”) Vex C-Channel x2
- 15 hole long 3 hole wide (7.5” x 1.5”) Vex C-Channel
- 5 hole long 3 hole wide (2.5” x 1.5”) Vex C-Channel x2
- 3 hole long 3 hole wide (1.5” x 1.5”) Vex C-Channel
- 12 hole long 2 hole wide (6” x 1”) Vex C-Channel
- Vex 12 hole (6”) linear slider

Gears

- Vex 84 tooth high strength gear x2
- Vex 60 tooth high strength gear
- Vex 36 tooth high strength gear (to be modified so that it is a 23 tooth slip-gear)
- Vex 19 tooth rack gear x2
- Vex 12 tooth high strength pinion
- Vex 12 tooth low strength pinion

Misc

- Array of Vex standoffs
- Screws, nuts, collars, zip ties, electrical tape, bearings, and spacers

Electronics

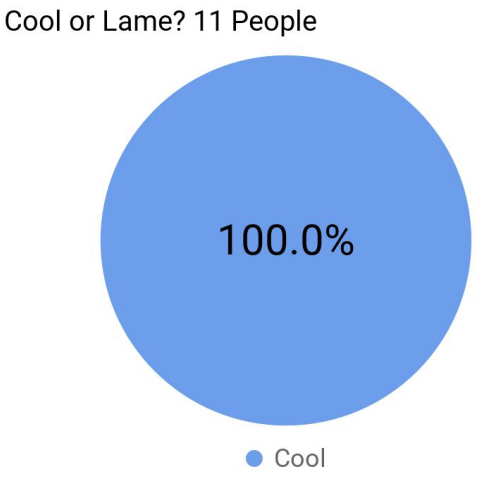
- Vex 393 torque geared motor x3
- Arduino UNO (Microcontroller that does the ‘thinking’)
- Pixy CMUcam5 Sensor (Camera that processes image data then sends processed information to the arduino UNO)
- Arduino Motor Shield (Contains integrated relays so that I can power the vex motors with arduino control but with a separate 7.2 volt battery)
- Relay for puncher motor.
- Various wires
- Optional sensors for greater user control

Product / Test Analysis

It seems pretty viable for robots next year to be able to use Pixy cameras to lock on to and fire at flags. I have very little doubt that among the Vex competition community this idea will become widespread and widely used this season. Though I wouldn't be able to monetize it among them, Vex already is. I don't really care about monetizing this design, though I could, because people love sentry guns. The major setback to selling these things would be the cost to make one, which is probably a little over $150, the Pixy camera is $70, the motors and battery are probably $20 each knowing Vex, though I could probably find a similarly effective products, I must re-state that my objective is not to sell this but to test whether an idea is possible, and this test project shows that it is.

I embarked on a project that would test my skills in electronics, mechanical design, and code. I needed to first design a turret that would be able to pan and tilt with a compact firing mechanism. This required me to use my robotics experience to use large, pro-torque gear ratios so that the motors would be able to make precise movements and hold their position without running a PID loop. I had to make a custom indefinite rotation turntable and stable joints for the actual pivoting parts. On top of all of this I had to make it attractive looking, of course, which I personally think went pretty well, excluding the box housing the electronics and the duct-tape Pixy camera mounting.

As you can see from the Pie chart to the right, everyone thinks that color tracking turrets are cool. Especially the robotics club that is considering using them next year. Who isn't fascinated by a little turret that is more interested in them than any person in the right mind would be? Turrets like this appeal to people by seeming "alive" in a way that can be comforting, amusing, or terrifying, depending on whether or not you believe in the AI takeover doomsday scenario.



Test Results:

Camera refresh rate: The update speed on the Pixy camera is about 50 times per second, plenty for my needs in robotics.

Camera tracking: Other than the need to calibrate before use, after this is done the camera tracking is pretty darn good, reliable too.

Camera tracking distance: Can detect a 10” by 6” rectangle from twelve feet away.

Issues:

Black spots appearing in the camera’s vision which could potentially cause the camera to lose sight of a target if aligned in an unfortunate way. I haven’t found anything online about this yet but I haven’t searched too hard for anything.

I’m still unsure if it is possible to calibrate the Pixy for different light levels / environments without using the camera’s button (unreliable) or the Pixymon software. Ideally I would have a little strip of color in the Pixy’s view that I could tell the Pixy to set as the signature that it should be tracking. Currently I have to manually re-calibrate for different light settings, which is a bit of a pain. Perhaps I could also have a flashlight strapped to it so that the amount of light reflected is constant, so long as that doesn’t create glare, which is another problem, when reflective surfaces are at angles where glare makes them appear as white on the Pixy, it obviously won’t detect the color behind.

Key Contributors

I worked alone.

I would like to thank my mother for purchasing the Pixy camera, my friend Uday for the third relay, my other friend Koji for bringing back my Arduino UNO on such short notice, Mr. Kaehms for that other relay, Ms. Chou for the Arduino UNO with a good SPI connector, Mr. Brown for allowing me to get parts from the robotics portable on multiple occasions, my father for the cardboard box and access to a soldering iron, my little brother for volunteering to get shot at, and the robotics club for allowing me to use the VEX parts to build this bad boy.

References

"2018-19 VEX Robotics Competition Game." *VEX Robotics*, 28 Apr. 2018, www.vexrobotics.com/vexedr/competition/vrc-current-game. Vex game information.

"Arduino AR-15 Sentry Gun! Widget52!" *YouTube*, YouTube, 4 Aug. 2015, www.youtube.com/watch?v=0mlG7rN-inI.

"Arduino Motor Shield Rev3." *Arduino Uno Rev3*, Arduino , store.arduino.cc/usa/arduino-motor-shield-rev3. What ports the motor shield uses.

"CMUcam5 Pixy." *Overview - CMUcam5 Pixy - CMUcam: Open Source Programmable Embedded Color Vision Sensors*, Charmed Labs, www.cmucam.org/projects/cmucam5/wiki/Arduino_API. Arduino commands for Pixy.

"CSM Talons - VEXU - Innovate Award Winning Video." *YouTube*, YouTube, 19 Mar. 2017, www.youtube.com/watch?v=4lopzs6y3Kc. Pixy used in Vex, Starstruck

"How to Set Up a 5V Relay on the Arduino." *Circuit Basics*, 23 May 2018, www.circuitbasics.com/setting-up-a-5v-relay-on-the-arduino/. How to wire up a relay.

"Rotary Turret with Pixy CMUCam5." *YouTube*, YouTube, 8 Sept. 2015, www.youtube.com/watch?v=7Iz8BCHiy0w.